IN THE SPECIFICATION:

Please replace paragraph 1 at page 23, with the following rewritten paragraph:

To an output of each of the registers, r0-rn, a read port 16 is connected. The read port 16 is connected through the data signal line L20 to the ALU12, the latch circuits 58 and 511, the selectors 56, 57, and 59, etc. shown in FIG. 1, to read data (DATA) from a register, ri specified on the basis of the read address Ar. The read port 16 is connected via the address line L14 to the selector 56 shown in FIG. 1. This is done so in order to supply the read execution address Ar or the read address Ar. ~~It~~ The write port 15 is connected via the address line L13 to the selector 57 shown in FIG. 1. This is done so in order to supply the write execution address Aw or the write address Aw.

Please replace paragraph 3 at page 25, with the following rewritten paragraph:

Each of the operation instructions #I1 through #I4 indicates an expression by mnemonics, an expression by a machine language, and processing contents. Operation instruction #I1 shown in FIG. 4 is, in an instruction structure shown in FIG. 3A, add [r10], r11 represented by 5001400Bh in the machine language and has such contents that to a value of register, r10 of a register number indicated by r10 of the register array 11, a value of register r11 of a register number indicated by ~~r10~~ r11 is added and a result thereof is stored in the register r10 of the register number indicated by r10.

Please replace paragraph 3 at page 35 continuing onto page 36, with the following rewritten paragraph:

Note here that the instruction-execution-controlling portion 50 shown in FIG. 1 is constituted of the execution state machine 51, the instruction read state machine 52, the selector 53, the program counter 54, the incrementer 55, the selector 56, the selector 57, the inputting

selector 59, and the latch circuits 58, 510, and 511. In this embodiment, the register array 11 is constituted with, for example, the allocated cell blocks ~~SE1~~ SE13, SE14, SE23, and ~~SE43~~ SE24. To constitute 8192 registers, r0-rn, the memory cells MSE of these cell blocks SE1, SE14, SE23, and SE43 are used. The memory cells MSE are allocated so that the register array 11 may be provided with plural registers, r0-rn holding arbitrary values based on the write address Aw and the write control signal Sw and output the values based on a read address Ar.

Please replace paragraph 2 at page 37, with the following rewritten paragraph:

The selector 56 is constituted by allocating the cell block ~~SE33~~ SE34 etc. to it. To provide wiring between these and other circuits, the column wiring lines CO3 and CO4 etc. are used for interconnection. The selector 56 selects either one of a read execution address Ar to select one register or a read address Ar to select this register again. The selector 57 is constituted by allocating the cell block ~~SE34~~ SE33, etc. to it. To provide wiring between these and other circuits, the column wiring lines CO2 and CO3 etc. are used for interconnection. The selector 57 selects either one of a write execution address to select one register, ri or a write address Aw to select this register again.

Please replace paragraph 2 at page 48 continuing onto page 49, with the following rewritten paragraph:

The operation-processing device 100' has, for example, an instruction decoding~~/restoring~~ portion 3, a storage portion 4, a register array 11', and an instruction execution/operation portion 50'. The register array 11' is a set of plural registers. The storage portion 4 stores a compressed program AP for specifying these registers in the register array 11'. The compressed program AP is created by the program creation device 200 and used. For example, the compressed program AP is written using a ROM writer etc. into the storage portion 4 for storing a program prepared

in the operation-processing device 100'. This is because there is a case where as for the operation-processing device 100', the processor may be constructed of a programmable logic device (PLD) comprised of plural memory cells and arithmetic/logic elements. An occupation ratio of the memory cells which function as the ROM can be reduced. Of course, such a method may be employed as to manufacture the storage portion 4 for storing program separately from the operation-processing device 100', store the compressed program AP in the separate storage portion 4, and mount them on the same substrate. This is because as the storage portion 4, a read only memory (ROM) or an EEPROM (flash memory) is used.

Please replace paragraph 1 at page 49, with the following rewritten paragraph:

To the storage portion 4, the instruction decoding~~/restoring~~ portion 3 is connected, so that from this storage portion 4 a compressed program AP is read to decode a type of register and, based on this register type, the number of bits of an instruction to specify this register is restored. This is done so in order to provide a uniform length of instructions so that plural registers may be specified on the basis of this instruction. To the instruction decoding~~/restoring~~ portion 3, the instruction execution/operation portion 50' is connected, to which the register array 11' is connected. The instruction execution/operation portion 50' executes arbitrary operations by specifying a relevant register in the register array 11' based on a program having a predetermined instruction length restored by the instruction decoding~~/restoring~~ portion 3.

Please replace paragraph 2 at page 62 continuing onto page 63, with the following rewritten paragraph:

The following will describe an example of processing in the program creation system I. FIG. 19 is a table for showing an example of creating a program in the program creation system I. In FIG. 19, P1 is an image of a program-describing screen, P2 indicates contents thus

described, and P3 is a description of the corresponding conditions in the embodiment. This is because a compressed program may be created by editing, in the program creation device 200 shown in FIG. ~~1~~ 13, instructions to execute operation processing based on an operation program that performs the register-to-memory addressing processing based on a predetermined programming language.

Please replace paragraph 1 at page 63, with the following rewritten paragraph:

In the program creation device 200, on a display device 24 shown in FIG. ~~1~~ 13, the program-describing screen P1 based on the C language shown in FIG. 19 is displayed, thus creating a compressed program using the keyboard 22 and the mouse 23. In this case, data required to create the program for the microprocessor 101 is read from the database 21. For example, "global variable declaration", "function declaration", "local variable declaration", "substitution", "addition", "comparison", and "branch-off", which are necessary for creation of a program in the C language, are read. In this example, in a case where 32-bit registers, ri are used as many as N (=8192) in the microprocessor 101 related to a new design and manufacture, when the 8192 registers, ri are assigned serial numbers of 0 through 8191, "local variable declaration" is performed on registers, r0-r31 of a group comprising the 0'th through 31'st registers as a group having a higher use frequency.

Please replace paragraph 1 at page 74, with the following rewritten paragraph:

This instruction-bit-restoring decoder 13, for example, calls a subroutine shown in FIG. 26A to pick up a portion of instructions at step G1 of its flowchart and output the instruction signal S9 to the execution state machine 51. Besides, the instruction-bit-restoring decoder 13 shifts to step G2 to check on whether a relevant instruction pattern is of a jump instruction. If

this instruction pattern is of a jump instruction indicated by #F5, the process shifts to step G12 to output a flag condition and a jump address. Then, the process returns to step F1 of the main routine. If it is decided at step G2 that this instruction pattern is not of a jump instruction, the process shifts to step ~~G4~~ G3 to check on whether a code written in "register type 1" is "0" or "1" concerning this instruction pattern. If code "0" is written in "register type 1", the process shifts to step G4 to set the number of bits (m) of an instruction for "register No. 1" to five and take it out from the compressed program AP. Then, the process shifts to step G5 to add "0" of the eight bits over the m (=5) number of bits of the instruction for "register No. 1", thereby providing a 13-bit length. Then, the process shifts to step G7 of the flowchart shown in FIG. 26B.